04 | 存储：区块链的存储设计有定式吗？

2021-07-28 赵铭

《说透区块链》 课程介绍 >

讲述：赵铭
时长 16:32 大小 15.15M

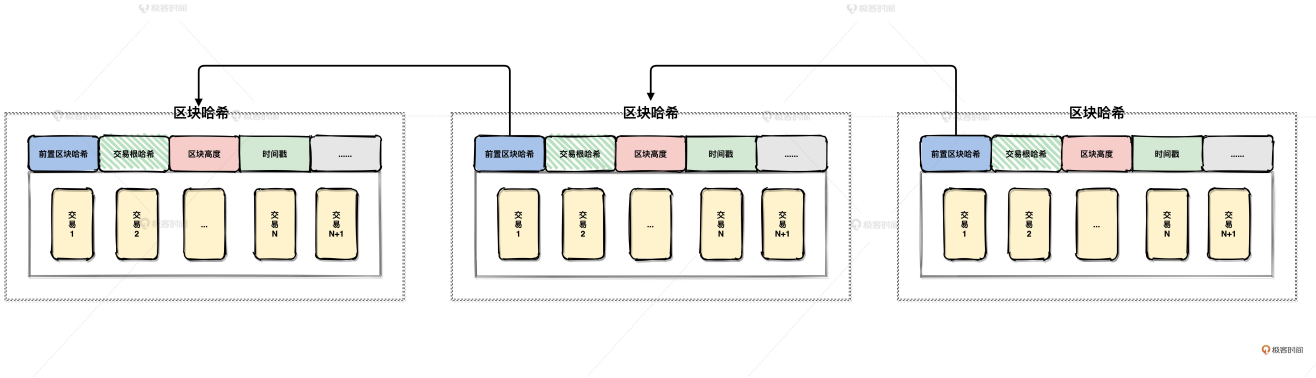
你好，我是赵铭。

上一讲我们从直观的角度，用“铁索连环”的例子和你探讨了区块链技术特性，也顺便引出了区块链技术基础。从今天开始，我会用几讲的时间为你解释每一种技术在区块链中的最核心应用，以点带面，为你勾画出区块链技术体系。

这一讲我会带你深入单个区块链节点，让你了解区块链存储是如何设计的。其实一说到存储设计，我们首先想到的是区块链里如何存储数据，以及要使用哪种数据库这些常规定式内容。但在我看来，这些都只拘泥于存储设计的表面。

想要真正把握住区块链存储的要点，我们更需要把握的是交易、区块还有状态这三个基础概念，有了这些基础，你再去分析区块链存储设计就会驾轻就熟了。

这里我为你准备了一张区块链的存储示意图，现在你可能还不太理解图里的逻辑关系，不过别担心，学完这节课之后，你就能真正理解后面这张图了。



区块链存储示意图

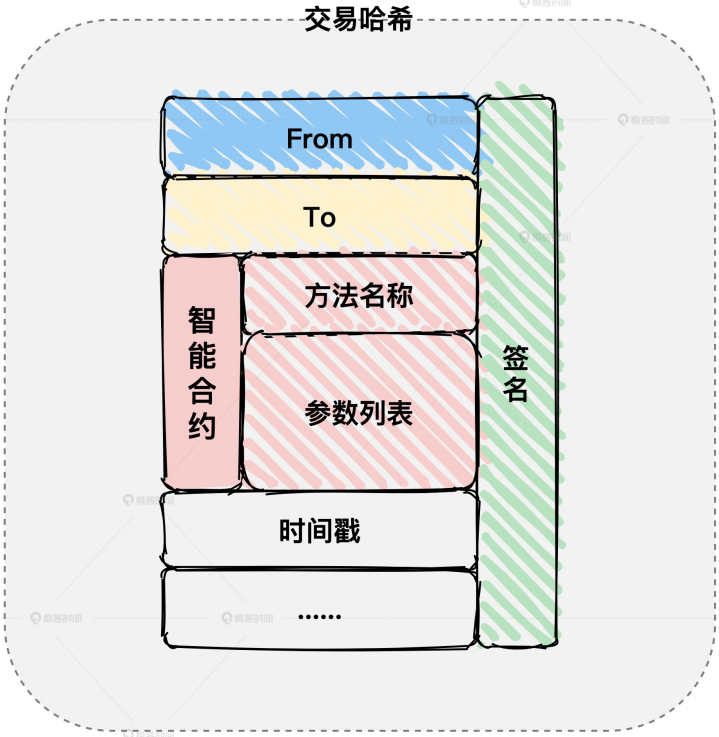
交易

我们首先需要理解的概念就是交易（Transaction），这是区块链中的最小也是最核心的知识点。因为我们最初接触区块链大多从比特币开始，所以我们通常会把交易理解成转账，但其实这个理解比较片面。其实在区块链中，交易的概念已经有所扩展。

从行为的角度解释，交易等同于操作（Operation），我们向区块链网络提交一笔交易实质上是发起了一个操作，而操作的具体内容与特定区块链协议有关，比如在以太坊中，一个操作可能就是执行了智能合约中的一个方法。

而如果我们**从计算机技术的角度分析，交易实质就是原子事物**，只是翻译的不一样，它们的英文都是 Transaction，交易是区块链网络中数据的最小的组成部分。一笔交易提交后，它只能有两种状态，要么成功要么失败，不可能存在成功了一半的情况。

不同区块链对交易的定义虽然一致，属性字段却有所差异，但却不影响我们抽象出一个通用的交易属性模板。需要注意的是，并不是所有的区块链都遵循下图中的规则，此处只是方便你理解交易主要有哪些属性。



交易属性示意图

从图中我们可以看到，一笔交易通常有 8 个属性（交易哈希本身也是一个属性）。From 跟 To 分别指向交易的发起方及接收方，这个很好理解，比如一笔货币的转账当然需要有转钱的跟收钱的。

跟智能合约相关的属性有三个，**智能合约**标识的是当前这笔交易要执行智能合约的名称，随后附带了执行智能合约对应的**方法**以及执行该方法时应该附带的**参数列表**，不同方法可能有着不同长度不同数据类型的参数，此处统一用参数列表表示。

接下来的**时间戳**字段表示该交易在客户端构建的时间。这个时间是客户端独立添加的，但是我们也不用担心该时间与标准时间的差异，因为区块链网络在接收这笔交易的时候，会有交易时间的校验，过早与过晚的交易并不会被网络接受，这在一定程度上限制了作假的可能性。

最后一个通用交易字段是**签名**，一般情况下是 **From** 字段的账户进行签发的，用于向网络证明这笔交易确实是这个账户构建的，而不是其他人伪造的，主要是使用账户拥有者手中的私钥对交易进行签名，而私钥只有账户拥有者持有。就像我们平常生活中使用的印章，只不过私钥被伪造的几率几乎没有，除非被盗。

有一点需要注意的是，**区块链中所有的交易基本上都是从区块链网络外发起的，区块链网络只接收交易而不生产交易，且不对交易做任何改动**。也就是说，交易在客户端构建出来以后就固化了。

因此我们可以使用交易内容的哈希值作为交易在区块链网络中的标识，且该标识不在交易的字段内容中，怎么理解呢？可以这样想，身份证可以代表你这个人，但身份证并不是你人的一部分。

另外，你可能还有这样一个疑惑，如果每笔交易都是客户端独立构建的，并没有与网络中的其他参与方进行协商，那这个**交易哈希**不会重复吗？

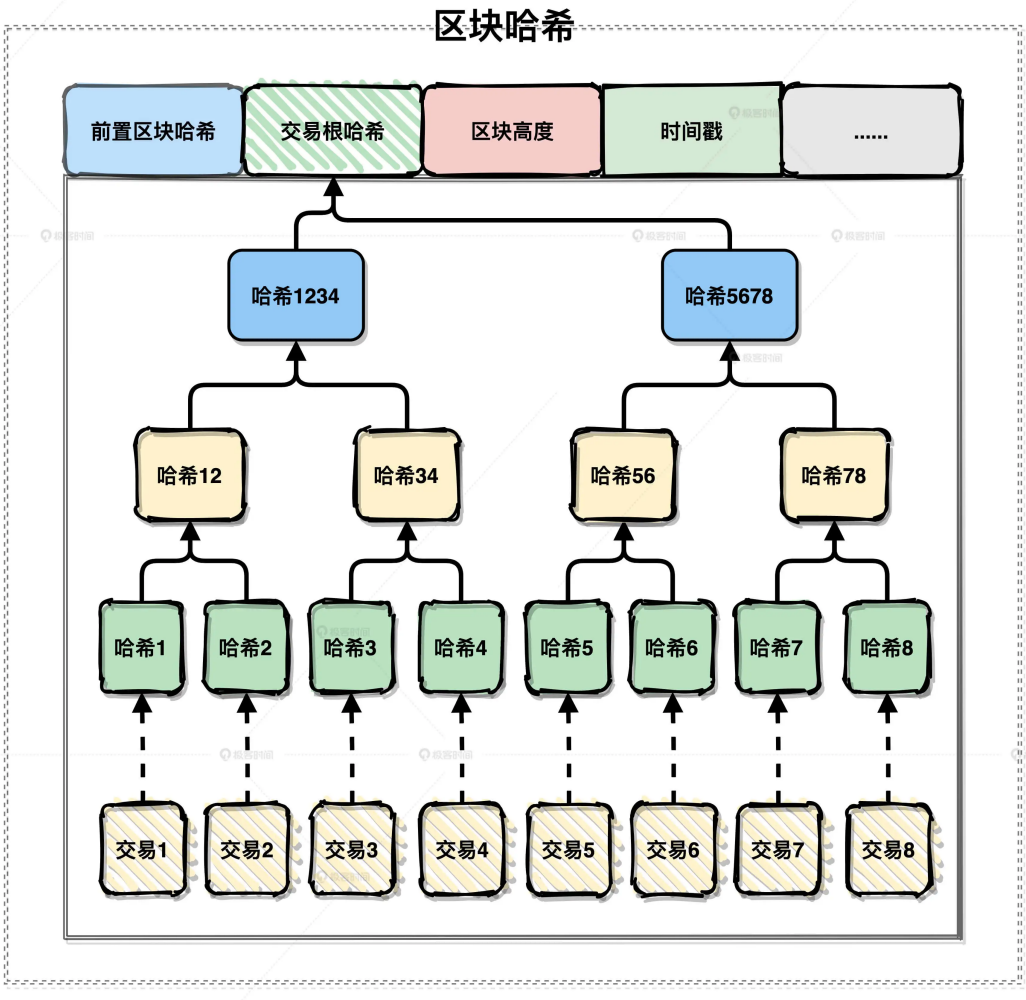
这里就要用到哈希算法的性质了，哈希重复意味着产生了哈希碰撞，在我们后续的密码学章节里，我们会讲到哈希碰撞的概率跟使用的哈希算法有关，且几乎不可能碰撞。

区块

理解了交易，我们再讨论一下，要用什么“容器”来存储这些交易数据。其实这个容器就是区块，你可以这样理解交易和区块的关系，交易相当于是货物，而区块则是能够容纳多个交易的集装箱。

在前面我们讲可追溯性的时候，我们提到区块链是时间段的数据前后关联依次串联整合成的信息链条，而每一时间段的数据我们就称之为区块（Block）。

区块是指将节点一段时间内收到的所有（有效）交易打包而形成的一种数据结构，之所以将有效扩起来，是因为有些区块链的设计也包括了无效交易。直接理解概念有些抽象，我们可以参照区块示意图去理解区块的数据结构。



区块示意图

从图中看，区块的设计貌似还比较复杂，不过你也别担心，其实我们只需要理清三个关键点就能领略区块的精髓。

区块结构

第一点我们需要理清的是区块的结构，从图中看，区块分成了区块头跟区块体。

区块头包含该区块的基础属性，重要的属性主要有 4 个：**前置区块哈希**用于区块间的关联，**交易根哈希**用于区块与交易的关联，**区块高度**用于标记当前区块在区块链中的位置，方便定位，而**时间戳**记录了区块打包的时间。

区块体则只有交易，且交易是有先后顺序的，一般是按照交易的时间戳字段进行排序。

区块间关联

第二个需要关注的点就是区块间的关联关系，这点我们已经多次提到，每一个区块都会包含**前置区块哈希**作为逻辑关联两个区块的那个锚点。

区块哈希跟交易哈希类似，是区块的外置属性，在区块构建完成后才能得到。如果我们从当前区块一步一步往前追踪，最终会找到创世区块，而创世区块也是有**前置区块哈希**的，只不过是一个空值。比如你可以通过以太坊浏览器查看创世区块的前置哈希。

以太坊创世区块前置哈希示意图

区块与交易

最后一点就是区块与交易的关联关系。虽然我们前面用了一个形象的比喻，把区块和交易想象成集装箱和货物，但我们并不明白其中的原理。从概念上理解相对复杂一些，这主要是因为引入了一个不常见的数据结构：**默克尔树**，我们可以先了解一下它。

默克尔树是一种树状结构，一般情况下至少有三层，分别是叶子节点，中间节点以及根节点，中间节点的层数取决于叶子节点的数量，叶子节点的数量越多，默克尔树的深度就越高。

它的构建逻辑是这样的：相邻的叶子节点进行哈希运算，得到的哈希值作为这两个叶子节点的父节点。然后同样的逻辑依次往上，最终倒数第二层仅剩的两个中间节点经过一次哈希运算得到他们的父节点，也就是整棵树的根节点，这样由哈希值构成的默克尔树就构建完成了。

对照前面的区块示意图我们可以发现，区块体中包含的交易所对应的交易哈希，可以作为默克尔树的叶子节点，然后依次往上进行哈希计算，最后得到的根哈希就是区块体所有交易的交易根哈希，这个数据将会记录在区块头中。

看到这里，你可能还有一个疑问，为什么要这么麻烦地引入默克尔树呢？为什么不直接把所有的交易揉在一起，取一个哈希就够了呢？

我们知道，对数据进行哈希计算得到的结果是可以作为数据指纹使用的，那也就意味着，哈希可以作为一种数据校验机制。

你可以想象一下，要是此时区块中有一笔交易被作恶者篡改了，如果我们设计**交易根哈希**的时候，仅仅对所有交易取一次哈希。那在数据校验不通过的时候，就很难找到被篡改的交易，尤其是在交易数量特别多的时候。

而如果我们使用了默克尔树，叶子节点哈希的任何变化都会传递至其父节点，一层层向上直到根节点，这就意味着根节点的值其实包含了所有叶子节点的哈希，但是却将可能被篡改的交易分开处理了，这样一旦出现问题，我们很容易就能判断出出错的分支。这提高了数据校验的灵活性，减少了很多不必要的资源浪费。

通过以上对区块逻辑上三个关键点的梳理，我们理清了区块设计的脉络。**区块链之所以叫区块链，从字面理解，正是因为区块这种特殊的数据结构决定的。**

状态

说完交易以及区块，我们再来理解一个经常被遗漏的概念就是状态（State），可能你之前从未听过，但它的作用却不可忽视。

在区块链中执行的每一笔交易都有一个输出，而**状态就是交易执行后输出的累积**。怎么理解呢？举个简单的例子：

> 2 + 3 + (4 * 7) + (8 - 9 / 3) + 23 = 61

“=”左边的每一个加法旁边的表达式我们可以认为是一笔交易记录，“=”右边的 61 则是交易执行后累积后的状态。

有限状态机

那么我们可以发现，即便丢失了表达式的结果，但是只要我们还记着表达式，我们就可以再重新计算出对应的值，而这就是**有限状态机**的概念，即**在一个封闭的系统中，如果状态起始条件一致，状态改变条件顺序一致，最终一定会得到一致的结果**。

而区块链就恰恰按时间顺序记录了所有的交易，因此即便丢失了状态，我们也可以很容易地重放状态，只要我们按照顺序再执行一次交易即可。所以从某种角度来讲，**区块链也是一种有限状态机**。

那你可能会问了，既然可以重放状态，为什么还要保留状态呢？我们不妨做一个情景假设，如果你现在正在执行一笔交易需要某个输入，而这个输入跟之前中间某个区块里某笔交易的输出有关。

如果不保留状态，你在执行这笔交易之前，就得重新执行相关联的交易，而那笔交易可能又与更早的另一笔交易有关，因此得一直回溯，直到找到源头为止。

所以从理论上来说，不保留状态是可以的，但是这就需要承担这种设计带来的相应后果。

与数据库的异同

如果你还是觉得状态这个概念不好理解，我们还可以从与数据库对比的角度来解释状态。我们与数据库进行 CRUD 交互的时候，在数据库中插入、更新、删除的记录就是所谓的状态，而你执行的每一条语句就是交易。换言之，如果你将从数据库创建到建表，再到插入数据、更新数据、删除数据等操作的所有 SQL 语句导出，换到其他地方，你也可以重放出一个一模一样的数据库出来。

区块链跟数据库都保存了历史操作记录跟状态数据集合。只不过**数据库更看重状态，而区块链却以记录历史区块为主，状态为辅。正所谓一个活在当下，一个怀念过去**。二者的逻辑并没有什么本质上的区别，只是侧重点不一样。

状态模型

了解了状态的设计理念，那你可能要问了，状态在区块链中是如何表现的呢？根据区块链的定位的不同，我们大致可以将状态模型的设计归纳为三种模型。

一种就是以比特币为首的专注于数字货币的区块链使用 UTXO 模型，即未花费的交易输出（Unspent Transaction Outputs）。在此模型中，每一笔交易都应该有 N 个交易输入，同时产生 M 个交易输出（N 与 M 可以不等）。

其中，交易输入是前序任意交易的未花费的交易输出。如果当前交易成交，该前序交易的输出也就变成了成交的交易输入，也就失去了再次成为交易输入的资格。UTXO 模型能够追踪数字货币的流向：**未花费的交易输入告知货币是从哪里来的，未花费的交易输出告知货币往哪里去**。

另一种就是以太坊区块链采用的账户模型，通过数字的加减表示账户余额的变化。

每一笔交易的执行，都会实现不同账户间余额的动态平衡，你我之间转账，你支出了 1 块钱，你的账户余额减 1，同时我的账户余额加 1，这种模型更符合我们日常生活的认知。同时，账户模型除表示余额以外，也支持自定义数据的存储，可以在基础账户之上衍生出智能合约数据存储。

而最后一类通用模型在账户模型的基础上更进一步，没有内置状态属性，可存储任意自定义数据，被广泛应用在联盟链中。联盟链的定位是支持企业级应用的区块链平台，而企业业务的种类及模式是无法预知的，因此无法在设计中内定状态模型。

既然众口难调，所以干脆将状态的设计工作留给企业应用开发者，状态可以自定义，而链本身只提供通用数据接口。

从三种状态模型的设计及应用场景看，**状态模型的选型没有唯一解，只要是能满足应用场景的模型设计就是好的模型**。

总结

这一讲我们主要深入到单个区块链节点，侧重了解了区块链存储的要点，主要讲解交易、区块两个数据结构及区块链状态模型。而我也并没有以具体的设计方案作为讲解重点，因为无论多么新颖，多么创新的区块链平台，其最基础的设计都逃不脱这三点。

区块链存储的设计没有定式，而只要你真正理解了交易、区块以及状态，你就是下一个区块链存储架构师。

1.交易是现实世界与区块链网络沟通的唯一媒介，是现实操作在区块链网络中的投影。

2.区块是交易的容器，由区块前后关联组成的链式结构是区块链名字的由来。

3.状态是历史交易输出的累积。

4.区块链是一种有限状态机。

说透区块链

赵铭 | 区块链服务平台资深架构师

识别二维码
免费试读

讨论

1. 通过这一讲，你是否对区块链状态有所共鸣？从你的角度来看，是区块更重要还是状态更重要，为什么？

2. 如果让你设计具体的区块链落盘方案，你会如何组织这三者的数据？

扩展阅读

关于默克尔树的详细解释，我推荐你阅读默克尔树究竟是棵什么树？这篇文章

欢迎你在留言区跟我互动，主动思考、积极交流会让你更有收获。如果这一讲对你有帮助，也欢迎你把这节课分享给自己的朋友、同事。

分享给需要的人，Ta订阅后你可得 20 元现金奖励

赞 3 提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 03 | 技术特性：用“铁锁连环”直观理解区块链特点

下一篇 05 | 密码学：为什么说区块链站在了巨人肩膀上？

精选留言 (5)

写留言

宇宙全栈
2021-07-28

我觉得对于开发者来说，状态更重要。区块好比框架，而状态是做具体业务时需要设计的数据结构和算法。

作者回复: 对的。状态更多的是服务于基于区块链的上层业务

1

姚莉莉
2021-08-01

我的理解：“数据库”记录我当前银行账户里有多少钱，比如有100万。“区块链”，可以拉流水，100万是怎么来的，哪些在何时的支出和收入。

不知道理解正确吗？请老师指正

展开

If_
2021-07-30

刚好去一个区块链公司，感谢sarah，好及时。

展开

作者回复: 希望能帮助到你！

高鹏0409
2021-07-29

存储结构可以和mysql类比吗？交易类似binlog，是有序操作记录；区块类似B+数存储；状态是经过多次修改后的值

作者回复: 是的，可以这样类比理解

静静聆听

2021-07-28

都重要，但区块保证了可追溯性，不可篡改性，是基础

展开

作者回复: 是的，没有交易跟区块，只有状态，这样的区块链跟数据库差异不大